CSP554—Big Data Technologies

Final Project Report

❖ Section I – Details

- **Project Topic** – To Build a ETL Data Pipeline using Big Data SQL Technologies on Google Cloud Platform.

- **Area of Project –** Formula 1 / Auto Racing Sport.

- **Dataset –** Formula 1 Dataset –

  The dataset is divided into Four csv files named circuits, constructors, drivers and driverGrid from year 1950 to 2022. Few of the important features are driverRef, points, position, wins, constructor_ref, start_position, constructor_positions, etc.

  https://www.kaggle.com/datasets/harrybassi13/formula-1

1) Problem Statement –

To build a data processing pipeline using Big Data technology to extract, transform & load the data, use various data techniques to find out interesting insights / analytics from the Formula 1 Dataset.

2) Approach for the Solution –

Data Ingestion: gcloud commands to move data to google cloud platform.

Database/Extract: Google Cloud Storage.

Data Transformation and Load: Google Dataflow.

Data Mining & Analysis: SQL (Google Big Query).

Visualize: Looker Studio by Google.

Section II – Literature Review

1) Google Cloud Platform

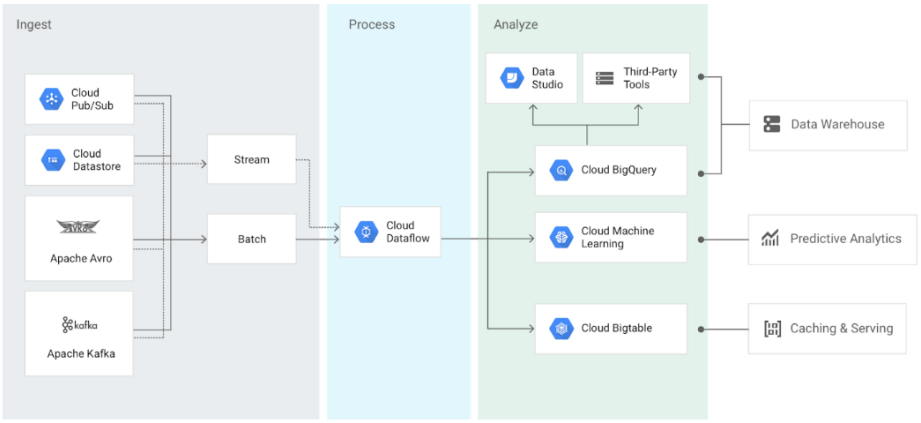
Google cloud platform is a public cloud vendor. Customers can access the computer resources held in Google's data centres worldwide via GCP or other public cloud vendors, either for free or on a pay-per-use basis.
GCP provides a full range of computing services, including tools for managing GCP costs, managing data, delivering web content and online video, and using AI and machine learning. You may manage your Google Cloud projects and resources using the web-based, graphical user interface provided by the Google Cloud console.

1.1) Benefits of Google cloud Platform

- Outstanding Availability and Uptime
- Live Migration of Virtual Machines
- Free Uptime Monitoring
- Leading Global Infrastructure
- Performance Optimization with Network Service Tiers
- Ease of Setup

1.2) Big Data on GCP

There are numerous services offered by Google Cloud Platform that address all common requirements for data and Big Data applications. All the services have their own advantages and disadvantages and relate to other Google Cloud products.



The place of Cloud Dataflow in a Big Data application

- **Tools using which we can do Big Data Analysis on Google Cloud Platform**

Google BigQuery - a data warehouse that processes and analyses large data sets using SQL.

Google Cloud Storage - Object storage which can store any amount of data.

Google Cloud Dataflow - is a serverless stream and batch processing service. We can build a pipeline to manage and analyse data in the cloud, while Cloud Dataflow automatically manages the resources

Google Looker Studio - offers interactive dashboards to build visual representations of data.

2) gcloud

The Google Cloud CLI is a set of tools to create and manage Google Cloud resources. We can use these tools to perform many common platform tasks from the command line or through scripts and other automation.

For example, we can use the gcloud CLI to create and manage the following:

- Compute Engine virtual machine instances and other resources
- Cloud SQL instances
- Google Kubernetes Engine clusters
- Dataproc, Dataflow clusters and jobs
- Cloud DNS managed zones and record sets
- Cloud Deployment Manager deployments

We can also use the gcloud CLI to deploy App Engine applications, manage authentication, customize local configuration, and perform other tasks.

3) Google Cloud Storage.

Google's response to an object store is cloud storage, A NoSQL database with high scalability. It automatically manages scaling using a distributed architecture. Instead of scaling with the size of the data collection, the queries scale with the size of the return set. Unpredictable data blobs can be shared, widely replicated, and versioned after being uploaded into a "bucket."

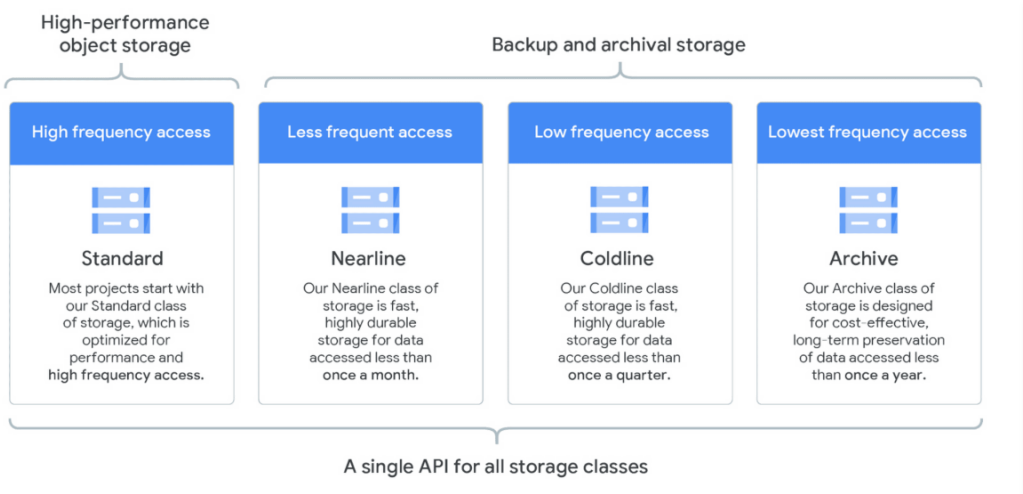
Any quantity of data can be stored and retrieved at anytime from anywhere in the globe using Google Cloud Storage. It can be utilized in a variety of situations, such as providing website content, archiving data for disaster recovery, or giving consumers direct download access to big data objects.

<u>Key features of Google cloud storage</u>:

- Provides unlimited storage with no minimum object size
- It is reliable and secure object storage option for users.
- Offers low latency and high durability

- Object lifecycle management: Cloud Storage allows users to define and assign conditions to a bucket that could trigger a data deletion or move to a less costly storage class.

Storage class: Google offers 4 types of storage classes for any workloads as per the user requirements:



- Benefits

- Scalability and flexibility
- Better collaboration
- Advanced security
- Data loss prevention
- Remote work made easy
- Life time storage

4) Google Dataflow

The serverless, quick, and economical Dataflow service provides both stream and batch processing. By automating infrastructure provisioning and cluster management, it gives processing jobs built in the open-source Apache Beam libraries portability and relieves operational burden from your data engineering teams.

ETL, batch, streaming processing, and many other types of data processing patterns can be created and implemented using Cloud Dataflow, a managed service. Data pipelines are constructed using dataflow. Python and Java jobs are supported by this service, which is built on Apache Beam.

Dataflow is a great choice for batch or stream data that needs processing and enrichment for the downstream systems such as analysis, machine learning or data warehousing

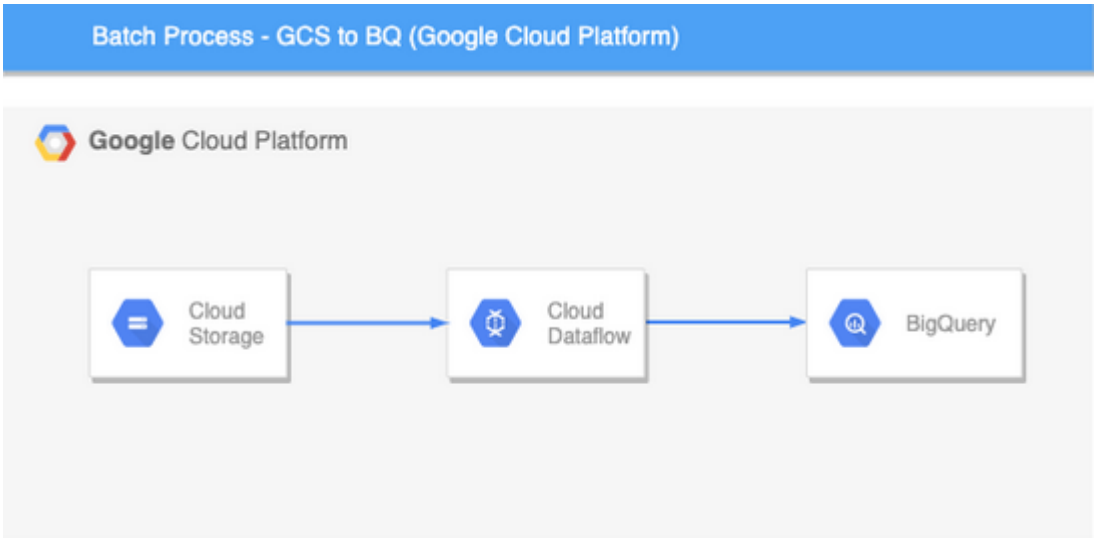
4.1) How does data processing work?

In general, a data processing pipeline involves three steps:
We read the data from a source (our case Google Cloud Storage), transform it (using DataFlow) and write the data back into a data lake (BigQuery Table).

a) Data is read into a PCollection from the source. Because a PCollection is intended to be distributed across numerous machines, the "P" stands for "parallel."

b) Then it performs one or more operations on the PCollection, which are called transforms. (In our case we convert the date into a format acceptable to the BigQuery Table format). Each time it runs a transform, a new PCollection is created. That's because PCollections are immutable.

c) After all of the transforms are executed, the pipeline writes the final PCollection to a BigQuery Table.



Once we have created your pipeline using Apache beam SDK in the language of your choice - Java or Python. We use Dataflow to deploy and execute that pipeline which is called a Dataflow job. Dataflow then assigns the worker virtual machines to execute the data processing, we can customize the shape and size of these machines. And, if the traffic pattern is spiky, Dataflow autoscaling automatically increases or decreases the number of worker instances required to run your job. Dataflow streaming engine separates compute from storage and moves parts of pipeline execution out of the worker VMs and into the Dataflow service backend. This improves autoscaling and data latency!

Key features:

- Autoscaling of resources and dynamic work rebalancing:
- Flexible scheduling and pricing for batch processing
- Ready-to-use real-time AI patterns

Benefits:

- Streaming data analytics with speed
- Simplify operations and management
- Reduce total cost of ownership

Pricing:

Cloud Dataflow jobs are billed in per-second, based on the actual use of Cloud Dataflow.

4.2) Apache Beam and how Dataflow uses it?

The Apache Beam paradigm offers practical abstractions that shield you from the rough aspects of distributed processing, like managing individual workers, partitioning datasets, and other similar activities. These minute aspects are completely handled by dataflow.

Beam is very helpful for tasks that require embarrassingly parallel data processing because they enable the problem to be divided into numerous smaller data bundles that can be treated concurrently and independently. Beam can also be used for pure data integration and extract, transform, and load (ETL) activities.

Some Basic concepts :

Pipelines
A pipeline is a collection of all the calculations necessary to take input data, transform that data, and write output data. It is possible to convert data between multiple formats by using input sources and output sinks that are the same type or of distinct types.

PCollection
The data for the pipeline is represented by a PCollection, which is a potentially dispersed, multi-element dataset. For each stage of your pipeline, Apache Beam transforms employ PCollection objects as inputs and outputs.

Transforms
A data transformation processing operation is represented by a transform. Almost any processing task can be carried out by a transform, including mathematical calculations on data, data conversion between formats, data grouping, data reading and writing, data filtering to output just desired elements, and data merging into single values.

Pipeline I/O
You can read data into your pipeline and write output data from your pipeline using Apache Beam I/O connectors for pipeline I/O. A source and a drain are the components of an I/O connector.

Runner

The software that accepts a pipeline and runs it is known as a runner. Most runners are massively parallel big-data processing systems' translators or adapters. For testing and debugging locally, other runners are available.

Source

Transformation that reads data from an outside storage device. Input data are often read from a source through a pipeline. You can alter the format of data as it passes through the pipeline since the source has a type that may differ from the sink type.

Sink

A transform that writes to an external data storage system, like a file or a database.

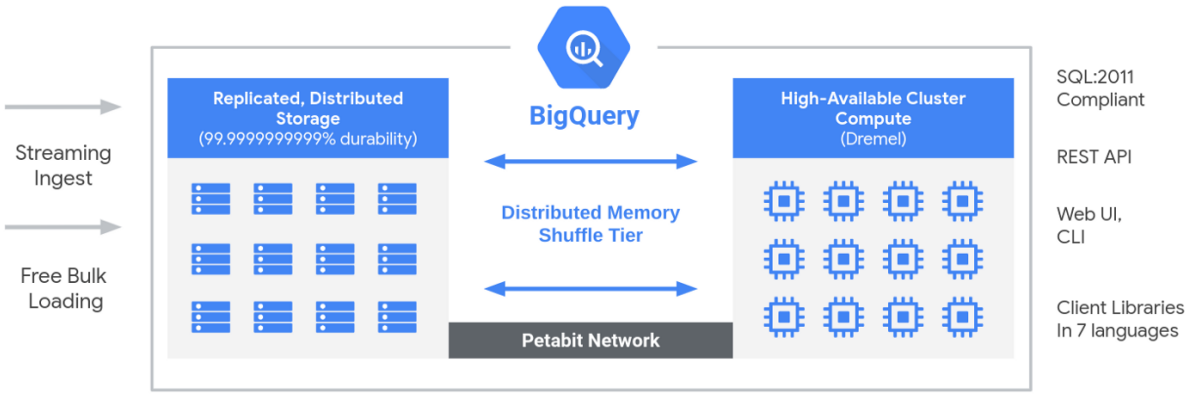
5) Google BigQuery

BigQuery is the cloud data warehouse component of Google Cloud Platform.
A serverless, scalable data warehouse called Google BigQuery makes it possible to analyze petabytes of data. Compared to on-premises servers, noisy data warehouses are better able to handle massive data collections. They enable real-time data access, allowing marketers and analysts to analyze data more quickly. Scaling up is simple and affordable, and it can handle more storage capacity.

Users can load data via batch or streaming loads, and it saves data in Google's Capacitor columnar data format. Use the traditional online UI, the web UI in the GCP Console, the bq command-line tool, or client libraries to import, export, query, and copy data.

Architecture

The serverless architecture of BigQuery separates storage from computation, allowing each to scale independently as needed. Customers benefit from this structure's enormous flexibility and cost control because they don't have to always maintain their pricey computational resources in operation. Compared to conventional node-based cloud data warehousing solutions or on-premises massively parallel processing (MPP) systems, this is considerably different. Additionally, this strategy enables clients of any size to upload their data into the data warehouse and begin performing Standard SQL analyses without having to worry about database administration and system engineering.



6) Google Looker Studio

Looker Studio is a free application that transforms your data into insightful, simple-to-read, shareable dashboards and reports. The lightweight modeling language called LookML is made available by Looker so that each organization's data specialists can specify their data in this manner. Thanks to LookML, which directs Looker on how to query data, everyone inside the organization may produce easily readable reports and dashboards to examine data trends. For the creation of original data apps and experiences, Looker offers additional capabilities.

Looker integration with Big Query:

Looker offers hosting on Google Cloud. Because Looker is platform neutral, it links to data in BigQuery as well as other public clouds. Looker is not necessary for using BigQuery. However, if your BigQuery doesn't offer these services, you might want to think about using Looker instead.

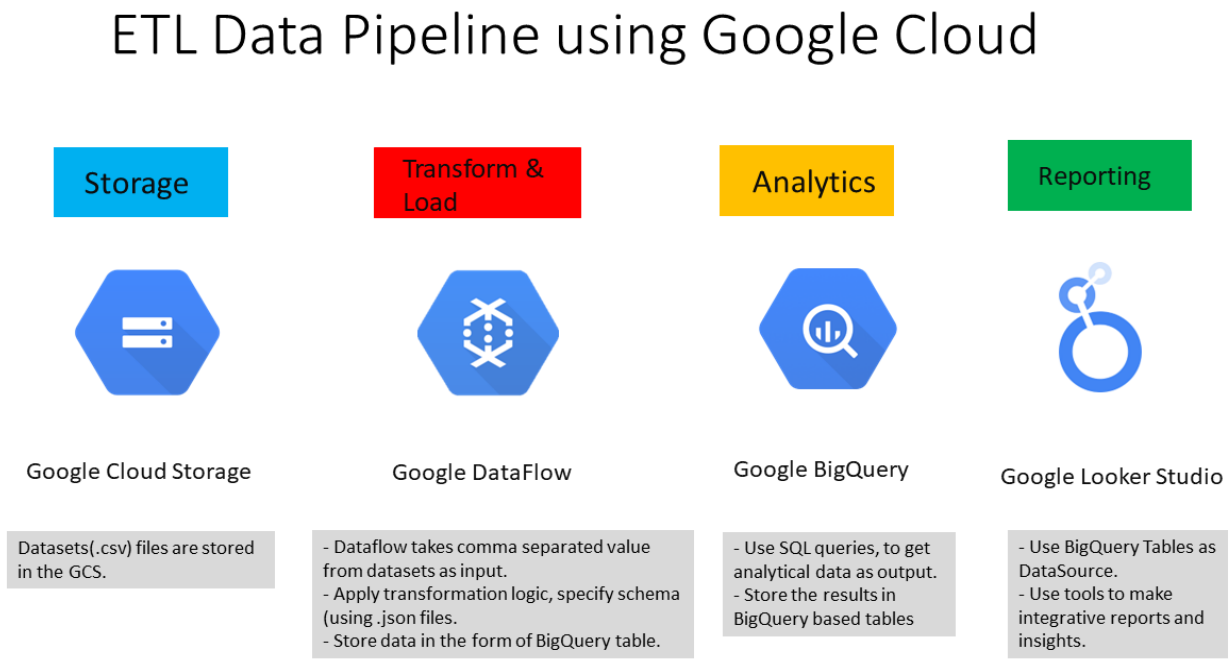
Things you can do with Looker studio:

- User can use tables, graphics, charts
- Change fonts and colors
- Brand the report with the client logo
- Add a video

The ability to import data from sources other than Google, such as Facebook Ads or Insights, LinkedIn Ads, or data from other sources, is another distinctive feature of Looker Studio. Since all of the reports are dynamic, if the primary data source is modified, the new or updated data will automatically display on all of the reports that rely on that source.

You can choose whether to grant users access so they can simply view the reports or the power to make changes by sharing the reports, which is another option.

❖ **Section III – Project Execution/Resultsx**



Step 1: In local window setting the google cloud project.

```
PS D:\> gcloud config list project
[core]
project = total-ensign-370100

Your active configuration is: [default]
PS D:\> gcloud config set project total-ensign-370100
Updated property [core/project].
```

Step 2: Copy Big-Data-Project.zip folder from local environment to google cloud platform using gcloud.

```
PS D:\> gcloud cloud-shell scp localhost:Big-Data-Project.zip cloudshell:
The server's host key is not cached. You have no guarantee
that the server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 3072 SHA256:o1rOK5kA5TbeZhY1yIS+0emjuPHFVIqBanFJ+51I0X4
If you trust this host, enter "y" to add the key to
PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without
adding the key to the cache, enter "n".
If you do not trust this host, press Return to abandon the
connection.
Store key in cache? (y/n, Return cancels connection, i for more info) n
Big-Data-Project.zip      | 538 kB | 538.3 kB/s | ETA: 00:00:00 | 100%
PS D:\>
```

Step 3: Now go to your Google Cloud Shell & initialize the project.

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to total-ensign-370100.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
anirali@cloudshell:~ (total-ensign-370100)$ gcloud config set project total-ensign-370100
Updated property [core/project].
anirali@cloudshell:~ (total-ensign-370100)$ ls
'~\.'                circuits.csv            '\dataset_example\'    dataset_example     README-cloudshell.txt
 Big-Data-Project.zip  dataflow-python-examples  '~\dataset_example\'  'dataset_example\'
anirali@cloudshell:~ (total-ensign-370100)$
```

Step 4: Install unzip if not already present on GCP.

```
anirali@cloudshell:~ (total-ensign-370100)$ sudo apt install unzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unzip is already the newest version (6.0-26+deb11u1).
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
```

Step 5: Unzip the Big-Data-Project.zip file on your GCP.

```
anirali@cloudshell:~ (total-ensign-370100)$ unzip Big-Data-Project.zip
Archive:  Big-Data-Project.zip
  inflating: Big-Data-Project/data_ingest_transform.py
  inflating: Big-Data-Project/data_ingest_transform_drivers.py
  inflating: Big-Data-Project/data_ingestion.py
   creating: Big-Data-Project/datasets/
  inflating: Big-Data-Project/datasets/circuits.csv
  inflating: Big-Data-Project/datasets/constructors.csv
  inflating: Big-Data-Project/datasets/driverGrid.csv
  inflating: Big-Data-Project/datasets/drivers.csv
   creating: Big-Data-Project/schema-json/
  inflating: Big-Data-Project/schema-json/circuits.json
  inflating: Big-Data-Project/schema-json/constructors.json
  inflating: Big-Data-Project/schema-json/driverGrid.json
  inflating: Big-Data-Project/schema-json/drivers.json
anirali@cloudshell:~ (total-ensign-370100)$ ls
'~\.'                Big-Data-Project.zip   dataflow-python-examples  '~\dataset_example\'  'dataset_example\'
 Big-Data-Project    circuits.csv           '\dataset_example\'        dataset_example       README-cloudshell.txt
anirali@cloudshell:~ (total-ensign-370100)$
```

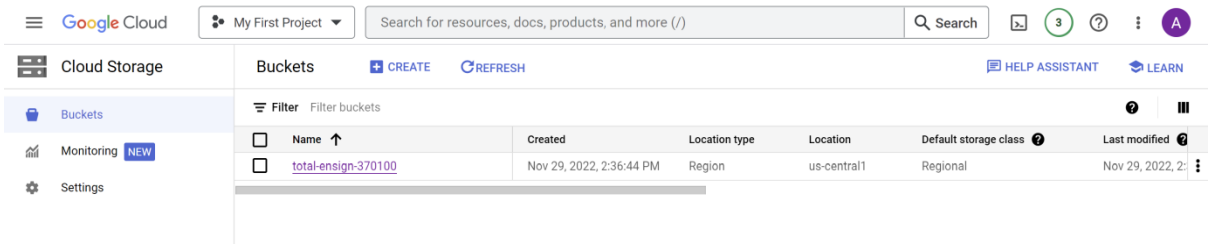
Step 6: Export your current Project on GCP to $Project.

```
anirali@cloudshell:~ (total-ensign-370100)$ export PROJECT=total-ensign-370100
anirali@cloudshell:~ (total-ensign-370100)$ gcloud config set project $PROJECT
Updated property [core/project].
anirali@cloudshell:~ (total-ensign-370100)$
```

Step 7: Create a Bucket on Google Cloud Storage. We are naming it after our Project Name.

```
anirali@cloudshell:~ (total-ensign-370100)$ gsutil mb -c regional -l us-central1 gs://$PROJECT
Creating gs://total-ensign-370100/...
```

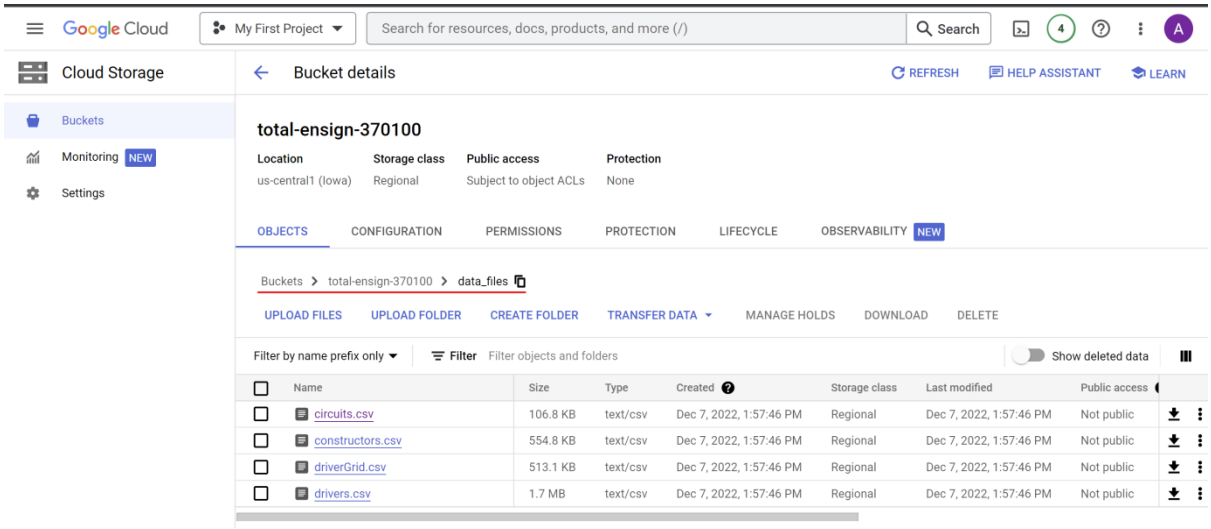
We can see under Cloud Storage that the Bucket is created using Project name as the Bucket Name.



Step 8 : Copy datasets (.csv files) from the Big-Data-Project/datasets folder to the Newly created Bucket /data_files/

```
anirali@cloudshell:~ (total-ensign-370100)$ gsutil cp ./Big-Data-Project/datasets/*.csv gs://$PROJECT/data_files/
Copying file://./Big-Data-Project/datasets/circuits.csv [Content-Type=text/csv]...
Copying file://./Big-Data-Project/datasets/constructors.csv [Content-Type=text/csv]...
Copying file://./Big-Data-Project/datasets/driverGrid.csv [Content-Type=text/csv]...
Copying file://./Big-Data-Project/datasets/drivers.csv [Content-Type=text/csv]...
/ [4 files][  2.9 MiB/  2.9 MiB]
Operation completed over 4 objects/2.9 MiB.
anirali@cloudshell:~ (total-ensign-370100)$
```

We should see all the .CSV files in the Google Cloud Storage.



Step 9: Make a BigQuery sink location named ‘Lake’.

```
anirali@cloudshell:~ (total-ensign-370100)$ bq mk lake
```

Step 10: Now we go to the Big-Data-Project folder,

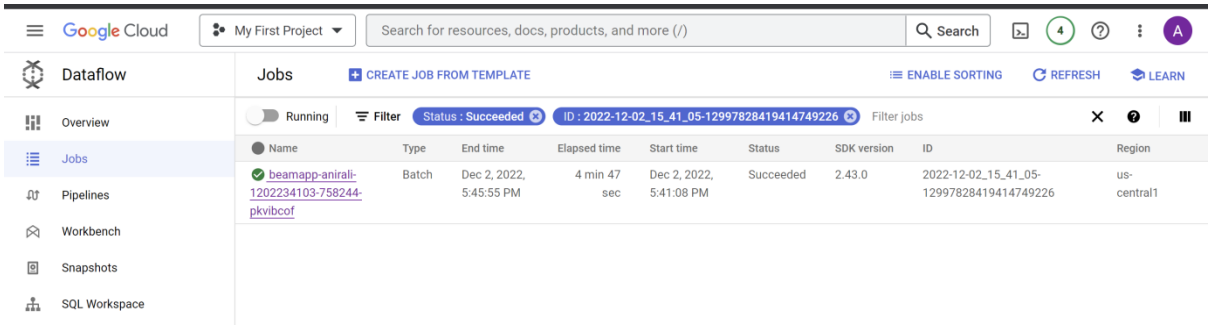
- Install a virtual environment.
- Install python3 package
- Activate it
- Install Apache-Beam[gcp]

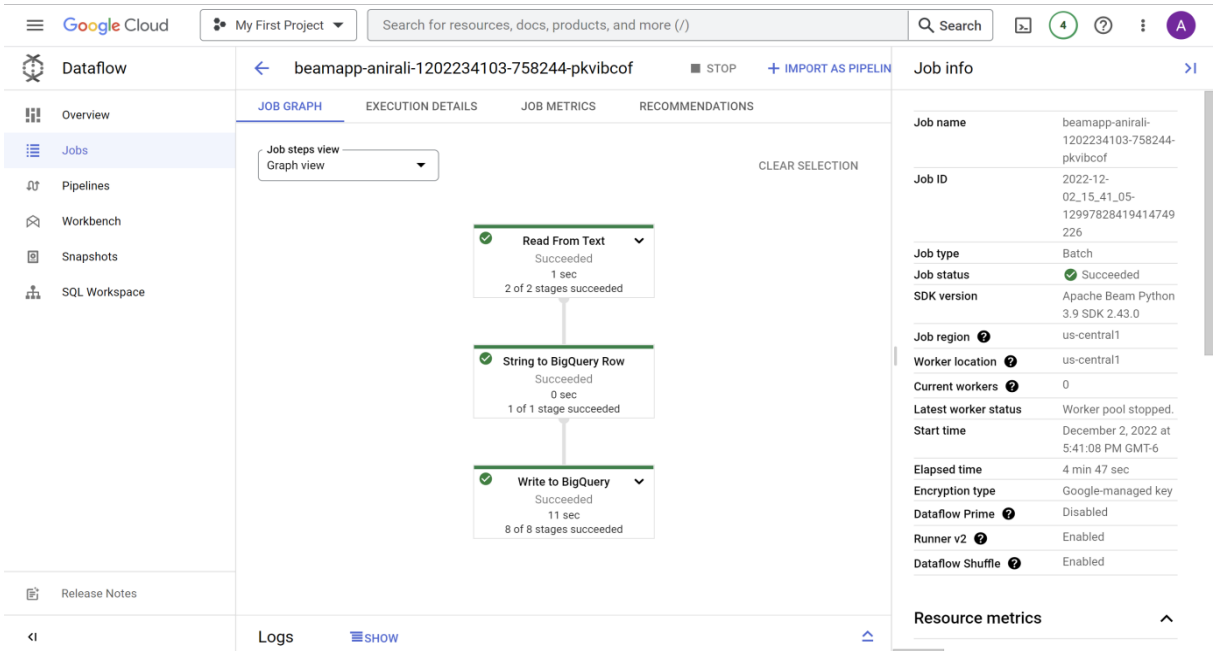
```
anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ sudo pip install virtualenv
Requirement already satisfied: virtualenv in /usr/local/lib/python3.9/dist-packages (20.16.6)
Requirement already satisfied: filelock<4,>=3.4.1 in /usr/local/lib/python3.9/dist-packages (from virtualenv) (3.8.0)
Requirement already satisfied: distlib<1,>=0.3.6 in /usr/local/lib/python3.9/dist-packages (from virtualenv) (0.3.6)
Requirement already satisfied: platformdirs<3,>=2.4 in /usr/local/lib/python3.9/dist-packages (from virtualenv) (2.5.3)
anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ virtualenv -p python3 venv
created virtual environment CPython3.9.2.final.0-64 in 1045ms
  creator CPython3Posix(dest=/home/anirali/Big-Data-Project/venv, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/home/anirali/.local/share/virtualenv)
    added seed packages: pip==22.3.1, setuptools==65.5.1, wheel==0.38.4
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ source venv/bin/activate
(venv) anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ pip install 'apache-beam[gcp]'
Collecting apache-beam[gcp]
  Using cached apache_beam-2.43.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (14.3 MB)
Collecting fasteners<1.0,>=0.3
  Using cached fasteners-0.18-py3-none-any.whl (18 kB)
Collecting pydot<2,>=1.2.0
  Using cached pydot-1.4.2-py2.py3-none-any.whl (21 kB)
Collecting httplib2<0.21.0,>=0.8
  Using cached httplib2-0.20.4-py3-none-any.whl (96 kB)
```

Step 11: Next we run the data_ingest_tranform.py file along with the given command line arguments specifying Project, region, runner, staging_location, temp_location and Input file from the Google Cloud Storage.

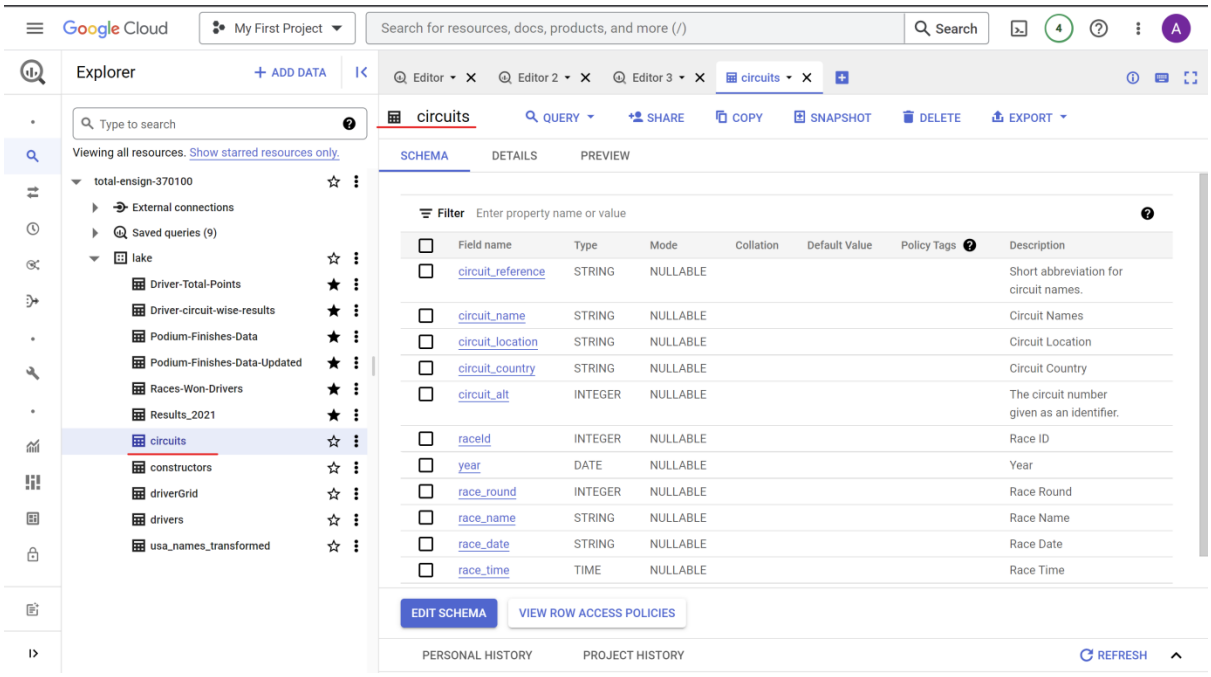
```
(venv) anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ python data_ingest_transform.py \
  --project=$PROJECT \
  --region=us-central1 \
  --runner=DataflowRunner \
  --staging_location=gs://$PROJECT/test \
  --temp_location gs://$PROJECT/test \
  --input gs://$PROJECT/data_files/circuits.csv \
  --save_main_session
```

This creates a Job under the Dataflow section.



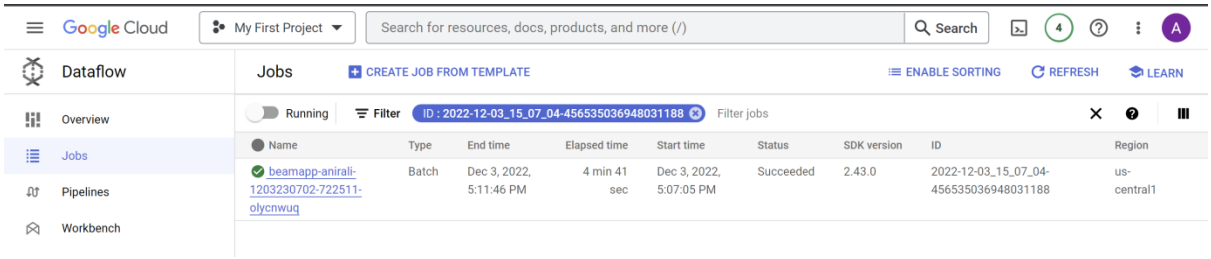


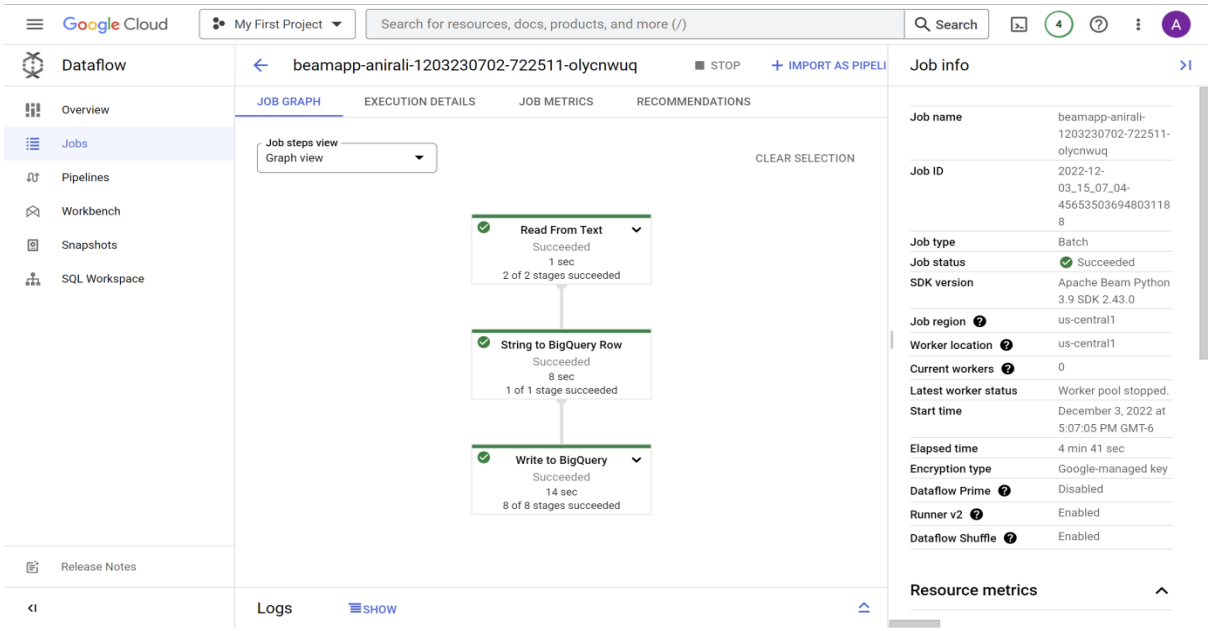
And we can see the Table created in the BigQuery, under the sink location named ‘Lake’.

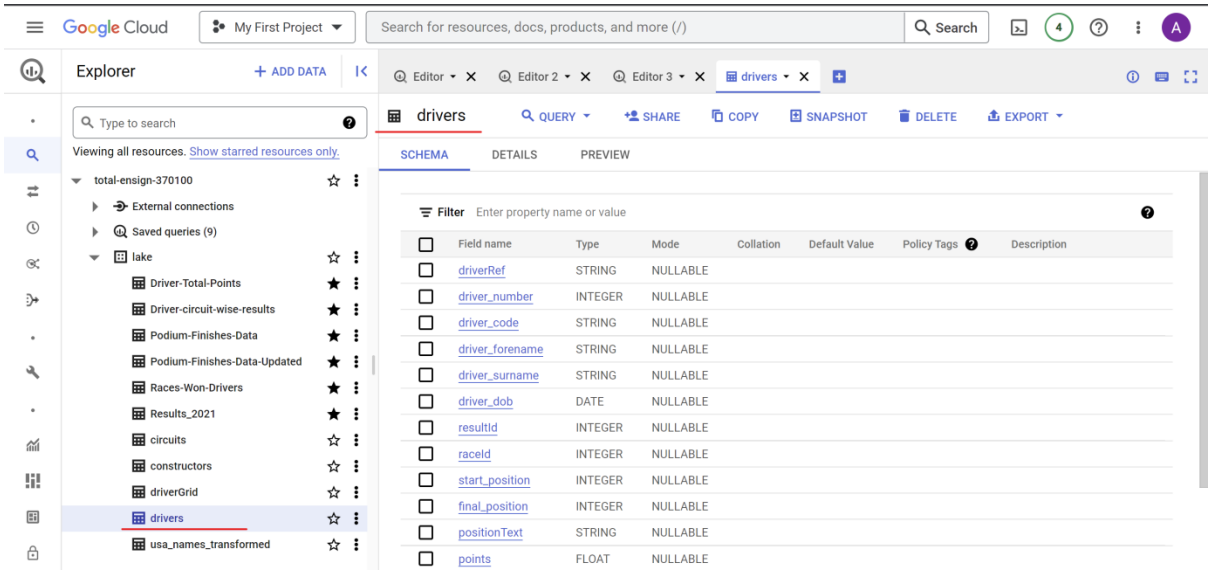


Step 12: Similarly, we run the data_ingest_transform_drivers.py file and get below results.

```
(venv) anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ python data_ingest_transform_drivers.py \
  --project=$PROJECT \
  --region=us-central1 \
  --runner=DataflowRunner \
  --staging_location=gs://$PROJECT/test \
  --temp_location gs://$PROJECT/test \
  --input gs://$PROJECT/data_files/drivers.csv \
  --save_main_session
```

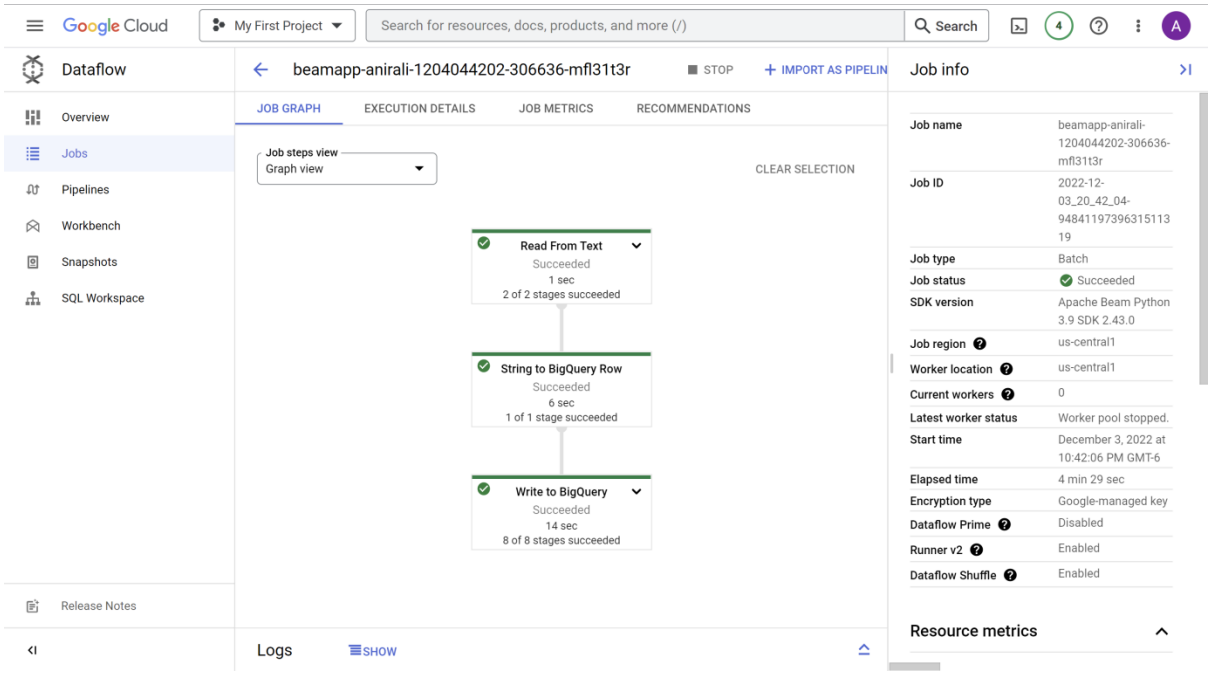


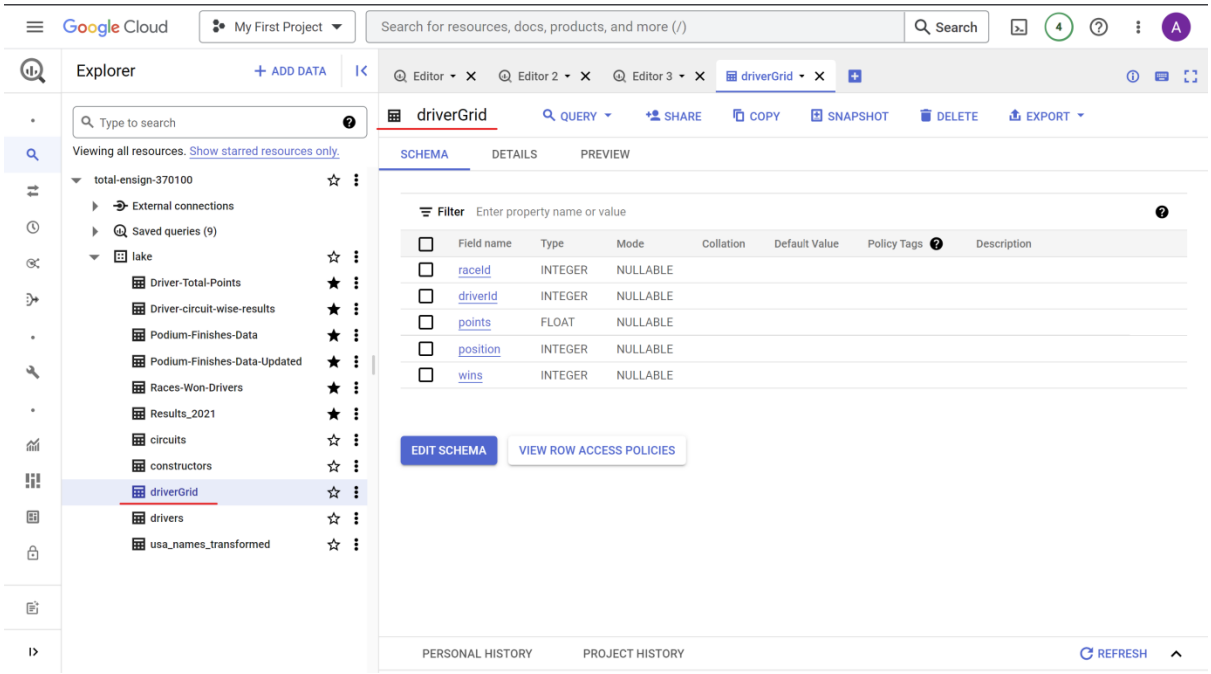




Step 13: Similarly, we run the data_ingestion.py file and get below results.

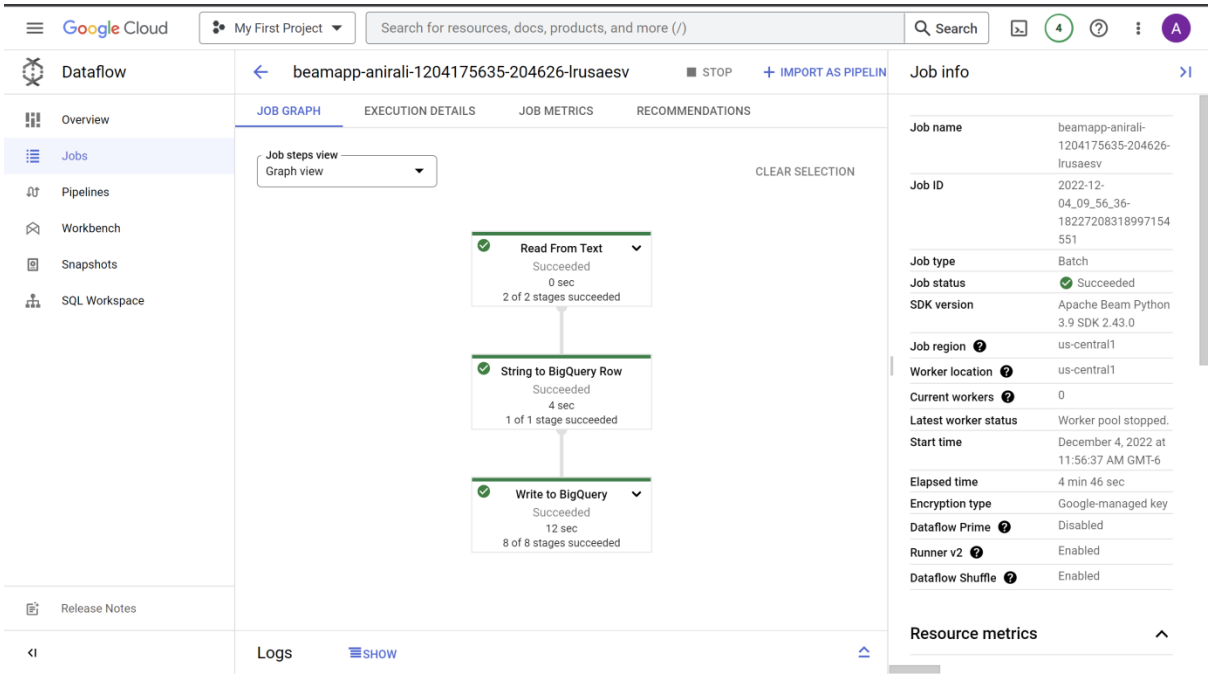
```
(venv) anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ python data_ingestion.py \
  --project=$PROJECT \
  --region=us-central1 \
  --runner=DataflowRunner \
  --staging_location=gs://$PROJECT/test \
  --temp_location gs://$PROJECT/test \
  --input gs://$PROJECT/data_files/driverGrid.csv \
  --save_main_session
```

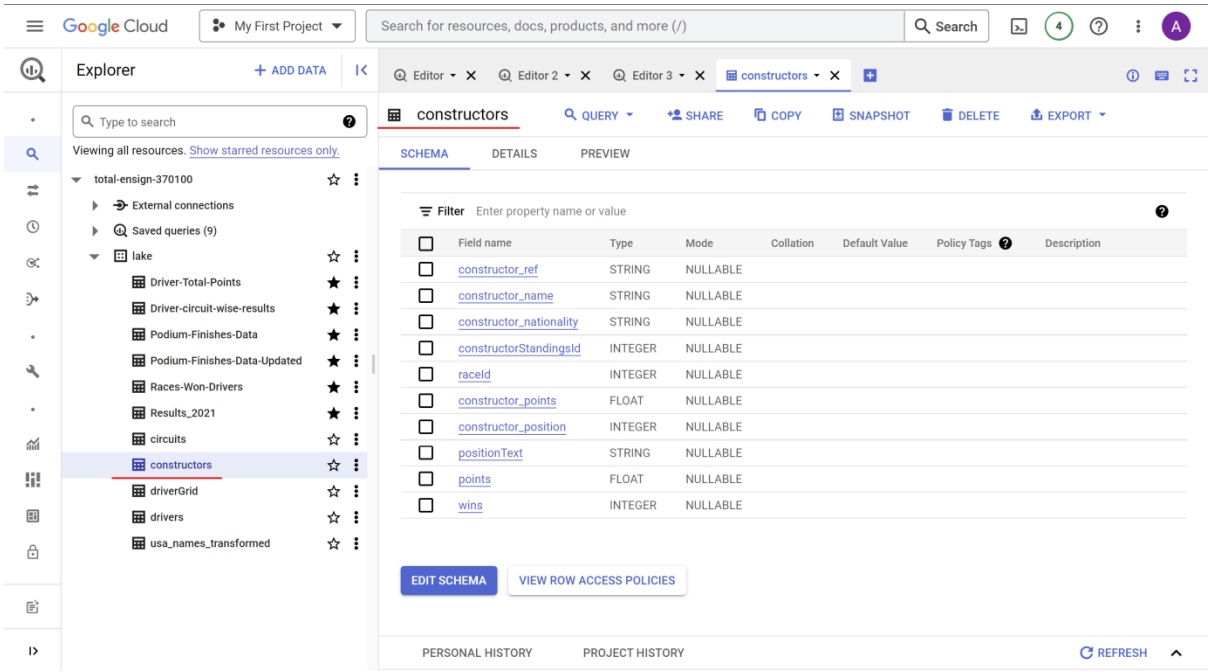




Step 14: Similarly, we run the data_ingestion.py file with the constructors.csv as the input and get below results.

```
(venv) anirali@cloudshell:~/Big-Data-Project (total-ensign-370100)$ python data_ingestion.py \
  --project=$PROJECT \
  --region=us-central1 \
  --runner=DataflowRunner \
  --staging_location=gs://$PROJECT/test \
  --temp_location gs://$PROJECT/test \
  --input gs://$PROJECT/data_files/constructors.csv \
  --save_main_session
```

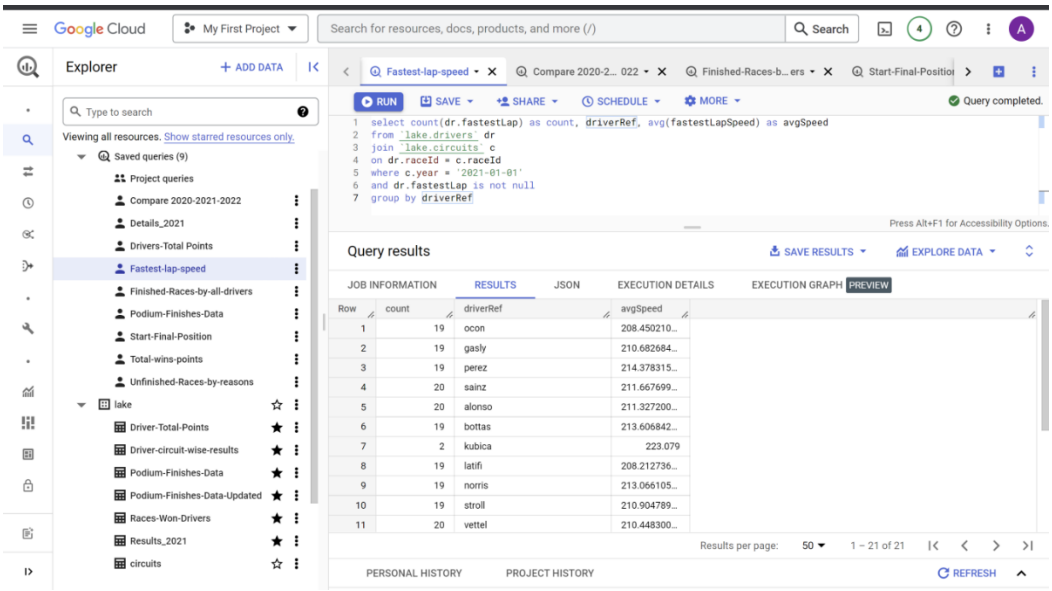




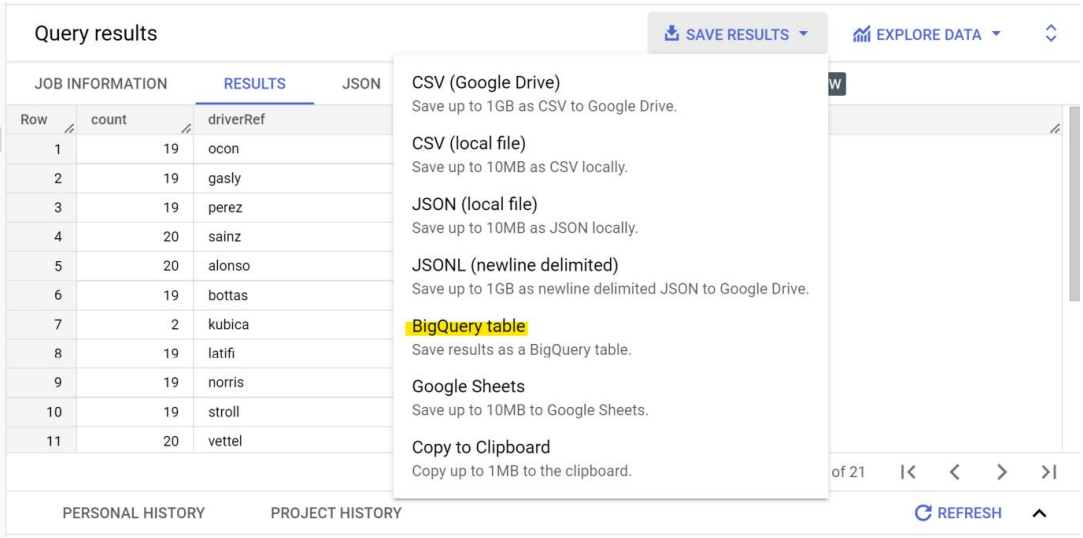
- Next, we write several SQL queries using the 4 tables which we created from the datasets.
- We store the Output of those queries as a BigQuery Table.
- We use these BigQuery Tables as an input source to make reports using various BarGraph, PieCharts, etc.

Below are the steps of how it Add **BigQuery DataTable** as **DataSource** to **Looker Studio** example of one Query.

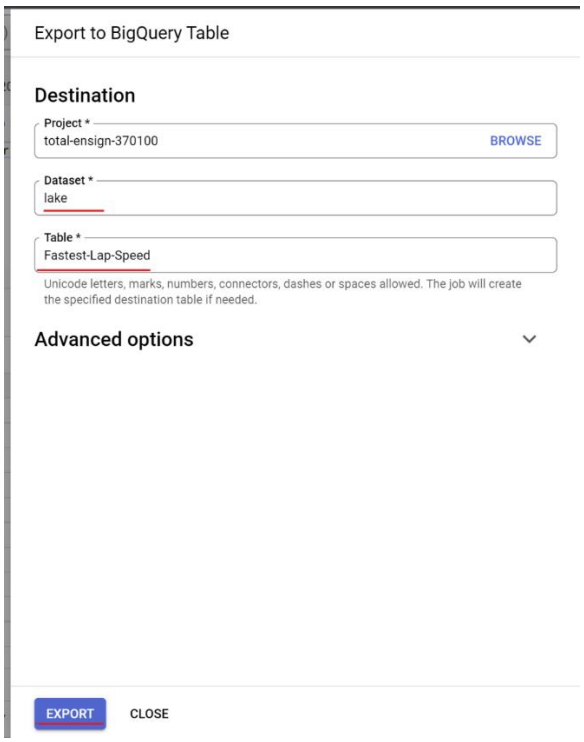
a) Write the SQL Query in BigQuery window to get interesting insights as a form of Output.



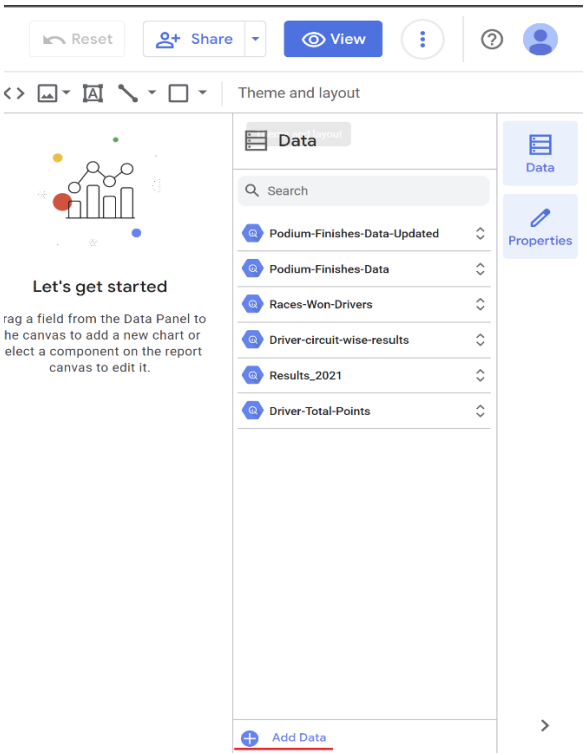
b) Click on Save Results option and select-> BigQuery Table option.



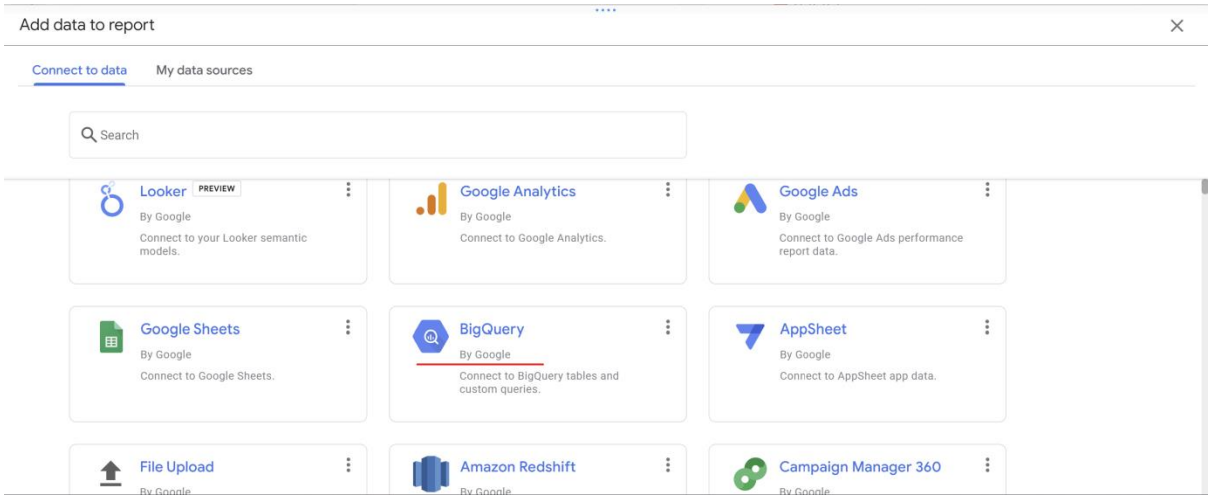
c) Select Project – Dataset – and give the Table Name and click on Export.



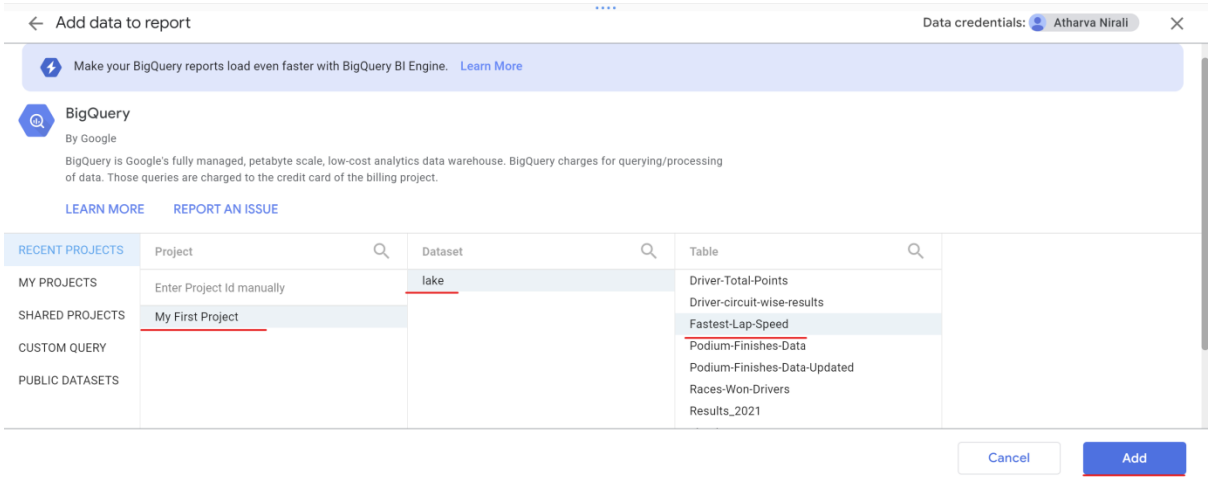
d) Next in the Looker Studio -> Click on Add Data.



e) Select the BigQuery Option.



f) Select the Project -> Dataset -> Table Name. This is the same Table which we saved on BigQuery.



g) We can see that the Data Source is add to Looker Studio, which Is used to Build reports on.

Results/ Visualization –

You can find the Report file by clicking on the below link. It will direct you to the Looker Studio, where you can see the entire visualization.

I have performed a deep analysis of F1- 2021 season.

Visualization Link – Ctrl + Click the below link.

https://datastudio.google.com/reporting/ee505c2a-f1a6-4f3c-89bc-5947f247c638

I have also uploaded the Visualization PDF file with the submission.

BigQuery SQL queries which are used to gather the data are below -

https://console.cloud.google.com/bigquery?sq=240469717449:f4bddea607814c5cbd19e22041107a23

https://console.cloud.google.com/bigquery?sq=240469717449:78a5348e641f464f836e976f93bf5ec8

https://console.cloud.google.com/bigquery?sq=240469717449:33c4fd751058456da72ed2f9176aace7

https://console.cloud.google.com/bigquery?sq=240469717449:58b424808e0043f1bab7c0110faa3ad2

https://console.cloud.google.com/bigquery?sq=240469717449:ad00cf028c1045729ef101610c166a1e

https://console.cloud.google.com/bigquery?sq=240469717449:135e6b7a7eaa49eb8aff93ddbc4fd024

https://console.cloud.google.com/bigquery?sq=240469717449:f2d11369a6ca495086853692e26497d2

https://console.cloud.google.com/bigquery?sq=240469717449:aa85c25a1e9d4b60a54057598ae28c36

❖ **Section – IV: Reference resources**

https://towardsdatascience.com/google-cloud-services-for-big-data-b9a657877ae2

https://cloud.google.com/dataflow

https://cloud.google.com/bigquery/docs/biglake-intro

https://cloud.google.com/data-fusion/docs/how-to/enable-transformation-pushdown

https://www.bounteous.com/insights/2021/06/29/benefits-using-bigquery-google-analytics-data

https://cloud.google.com/bigquery/docs/looker

https://www.kaggle.com/datasets/harrybassi13/formula-1

https://cloud.google.com/bigquery/docs

https://developers.google.com/looker-studio

https://cloud.google.com/dataflow

https://cloud.google.com/storage/docs

https://cloud.google.com/dataflow/docs/quickstarts/create-pipeline-python#local-terminal

https://cloud.google.com/dataflow/docs/guides/templates/provided-batch#cloud-storage-text-to-bigquery

https://cloud.google.com/dataflow/docs/guides/data-pipelines

https://cloud.google.com/storage/docs/creating-buckets

https://www.cloudskillsboost.google/focuses/3460?parent=catalog